ABSTRACT

The rapid evolution of artificial intelligence has made possible the development in speech recognition technology which is now actively penetrating almost every area of human life. Voice assistant is an important achievement in this field. A voice assistant is digital assistant that uses speech recognition, speech synthesis, and Natural Language Processing to provide user a required information or to perform any task like getting weather updates, getting traffic updates, creating remainder, sending mails, making calls and many other services just on a simple voice command. The system first uses speech recognition for converting the voice command into text for processing. Voice assistant relies on Natural Language Processing to resolve the barriers of understanding. After processing, voice assistant retrieves information related to the question using the knowledge base or various Application Programming Interfaces, if the command is task-based then the system calls are used. Finally, speech synthesis converts the output or result in speech format from the text. This system provides user hands-free access to various services. The result of a case study shows that the proposed voice assistant system can effectively utilize the user's command, Application Programming Interfaces or system calls depending on user requirement to provide the services.

KEYWORDS: Artificial Intelligence, Voice assistant, Speech recognition, Speech synthesis, Natural Language Processing

CONTENTS

ABBREVIATIONS

Abbreviations	Meaning
AI	Artificial Intelligence
NLP	Natural Language Processing
STT	Speech-to-Text
TTS	Text-to-Speech
DSP	Digital Signal Processing
API	Application Programming Interface
GUI	Graphical User Interface
URL	Uniform Resource Locator
IDE	Integrated Development Environment

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 Voice Assistant

Voice assistant is a task-oriented programming application or software that recognizes human speech and carries out commands pronounced by a user. It is powered by AI and bases its performance on cloud storage with millions of words and phrases in it. The rapid evolution of AI and machine learning made possible the development of speech recognition technology, which is how actively penetrating every area of our lives. And there's nothing to wonder about; for such social-dependent creatures as humans, speaking is a lot more natural activity than writing or of course, typing. An average human can type about 40 words in one minute, but pronounce 150. This contrast, alongside with many other benefits, vividly demonstrates why voice technology should be taken seriously. Voice assistants are not to be confused with virtual assistants, which are people who work remotely and can therefore handle all kinds of tasks. Rather, voice assistants are technology based. As voice assistants become more robust, their utility in both the personal and business realms will grow as well. The most common voice assistants in today's world are Siri by Apple, Google Assistant by Google, Alexa by Amazon, Samsung's Bixby, Microsoft's Cortana

1.2 History of Voice Assistants

Voice assistants have a very long history that actually goes back over 100 years, which might seem surprising as apps such as Siri have only been released within the past ten years. The very first voice activated product was released in 1922 as Radio Rex. This toy was very simple, wherein a toy dog would stay inside a dog house until the user exclaimed its name, "Rex" at which point it would jump out of the house. IBM began their long history of voice assistants in 1962 at the World's Fair in Seattle when IBM Shoebox was announced. This device was able to recognize digits 0-9 and six simple commands such as, "plus, minus" so the device could be used as a simple calculator. Darpa then funded five years of speech recognition R&D in 1971, known as the Speech Understanding Research (SUR) Program. One of the biggest innovations to come out of this was Carnegie Mellon's Harpy, which was capable of understanding over 1,011 words.

The next decade led to amazing progress and research in the speech recognition field, leading most devices from understanding a few hundred words to understanding thousands all with the help of AI technology. In 1994, Simon by IBM was the first smart voice assistant and really, the first smartphone in history. In 2008, when Android was first released, Google had slowly started rolling out voice search for its Google mobile apps on various platforms, with a dedicated Google Voice Search Application being released in 2011. This led to more and more advanced features, eventually leading to Google now and Google Voice Assistant. Then, this was followed by Siri in 2010. Developed by SRI International with speech recognition provided by Nuance Communications, the original app was released in 2010 on the iOS App Store and was acquired two months later by Apple. Then, with the release of the iPhone 4s, Siri was officially released as an integrated voice assistant within iOS. Since then, Siri has made its way to every Apple device available and has linked all the devices together in a single ecosystem. Shortly after Siri was first developed, IBM Watson is announced publicly in 2011. Watson was named after the founder of IBM, and was originally conceived in 2006 to beat humans at a game of Jeopardy. Now, Watson is one of the most intelligent, naturally speaking computer systems available. Amazon Alexa is then announced in 2015, helping with more accurate speech recognition. With Alexa Echo, the line of smart devices is announced to bring smart integration.

1.3 Overview of Proposed System

The Intelligent Desktop Voice Assistant is developed and designed to assist the user with the basic tasks. The proposed system take voice as input key from the user, this is done Speech recognition. The input is then processed by Natural Language Processing followed by Speech Synthesis for providing the required output in the form of voice, thereby providing hands free access to user. This conversational voice assistant is combination of both a task-oriented and knowledge-oriented workflow to carry out almost every task that user throw at it. A task-oriented approach workflow includes task like writing and sending the E-mail, while knowledge-oriented workflow includes answering ‘What is my current location?’ or solving the mathematical calculations. The system is designed in such a way that all the services provided by the computer devices are accessible by the end user on the user's voice commands. Voice assistant relay on uninterrupted internet connection for smooth user experience.

1.4 Motivation

Nowadays, user mainly uses voice assistant which comes inbuilt in the specific device. They are bound to use the services which are available in particular voice assistant but not which they prefer. Also, switching among different voice assistant ultimately results in switching manufacturer of that particular device. This process of switching voice assistant faces two main challenge, one is adapting to change in system environment and second is high cost requirement.

The proposed system is strongly inspired by the idea of overcoming these challenges by giving user more freedom in selecting and personalizing the voice assistant as well as the services without causing any change to system environment and that too in lowest cost possible.

1.5 Organization of Report

Chapter one contains the introduction of the proposed system which comprises information regarding the voice assistants, their history and overview. This chapter also describes the motivation and scope for the project.

Chapter two includes the literature review which refers to the study that has been carried out on different voice assistant systems over the previous years that have some relevance with existing system and also various technologies which are used for proposed system. This chapter also specifies the aim and objectives.

Chapter three explains the tools and technologies which are used for developing voice assistant. It also explains about the APIs.

Chapter four explains proposed approach and system architecture which includes the detail of how the voice assistant has been developed with different components.

Chapter five discusses the implementation details of proposed system, this chapter explains the coding part developed for performing various functions.

Chapter six discusses the results that were generated from the proposed system of voice assistant and showcases every necessary output needed to describe the proposed system precisely.

Chapter seven contains conclusion about the proposed system and it also describe limitation of study and future scope of proposed system.

CHAPTER 2
LITERATURE REVIEW

A literature review discusses published information in a particular subject area, and sometimes information in a particular subject area within a certain time period. It can be just a simple summary of the sources, but it usually has an organizational pattern and combines both summary and synthesis. A summary is a recap of the important information of the source, but a synthesis is a re-organization, or a reshuffling of that information.

2.1 Related Work

2.1.1 Voice Assistant for Home Automation

Prerna Wadikar, Nidhi Sargar, Rahool Rangnekar, Prof. Pankaj Kunekar (2020) "Home Automation using Voice Commands in the Hindi Language". The proposed project of Home Automation in Hindi language. Voice commands was to implement the dedicated hardware i.e. Arduino Uno and using voice recognition module that makes the system more cost-efficient and robust. The system can work on various connected devices like light, fan, AC, etc. This system allows users to make decisions and to regulate the home appliances with the help of voice assistants. **Chen-Yen Peng and Rung-Chin Chen (2018)** designed and built a tailor-made function for users without their attempt. Commands are taken from Google Home's voice recognition and Bluetooth signals are transferred to Raspberry Pi to control the connected devices. The proposed paper mainly focuses on researching combining characteristics of Google Home with Google Assistant Personal Voice Assistant using machine learning and thereby customizing this to meet the new needs of users.

2.1.2 Voice Assistant for Blind People

Steve Joseph, Chetan Jha, Dipesh Jain, Saurabh Gavali, Mapythnish Salvi (2020) proposed "Voice based E-Mail for the Blind". They designed the system that was helpful for sending emails for the blind people without the need of visual interaction with the screen. Speech-to-Text based Life Log System for Smartphones. The technique used in it was Microphone of Smartphone, STT (Speech-to-Text).

Nishank Tembhurne, Sumedh Vaidya, Afrin Shiekh, Prof. Swapnil Dravyakar (2019) developed a customized application "Voice Assistant for Visually Impaired People". This application is used to help the visually impaired to access most important features of the phone using text to speech and speech to text. The system had custom messaging feature, call log feature, notes making feature, OCR feature, web browsing feature, navigation feature in it. The custom app having these features made it possible for visually impaired users to do their basic things using electronic device without any other help.

2.1.3 Voice Assistant for Android and Linux

Shen Hui, Song Qunying and Andreas Nilsson (2012) proposed an "Intelligent Voice Assistant" application for androids. This project focused on the Android development over the voice control, relevant APIs and mobile device references ranging from Speech-To-Text, Text-To-Speech technology, Bluetooth headset support and camera; advanced techniques of Cloud computing and Multi-threading. This proposed system was developed by various technologies like Java, Android development, MySQL and Network connection technologies. **Harkishen Singh, Muskan Khedia, Jayashree Panda, Subham Mishra, Ankit Singh (2020)** developed the "Jarvis: The Personal Linux Assistant". The system aimed to provide the Linux user with virtual assistant that not only aid in their daily routine but also help in various automation activities. They further developed the complete server assistant by automating the entire server management process – deployment, backups, auto-scaling, logging, monitoring.

2.2 Artificial Intelligence

In computer science, the term artificial intelligence (AI) refers to any human-like intelligence exhibited by a computer, robot, or other machine. In popular usage, artificial intelligence refers to the ability of a computer or machine to mimic the capabilities of the human mind; learning from examples and experience, recognizing objects, understanding and responding to language, making decisions, solving problem and combining these and other capabilities to perform functions a human might perform. The ideal characteristic of artificial intelligence is its ability to rationalize and take actions that have the best chance of achieving a specific goal. A subset of artificial

intelligence is machine learning, which refers to the concept that computer programs can automatically learn from and adapt to new data without being assisted by humans. Deep learning techniques enable this automatic learning through the absorption of huge amounts of unstructured data such as text, images, or video. After decades of being relegated to science fiction, today, AI is part of our everyday lives. The surge in AI development is made possible by the sudden availability of large amounts of data and the corresponding development and wide availability of computer systems that can process all that data faster and more accurately than humans can.

Artificial Intelligence is divided into two different categories: weak and strong. Weak artificial intelligence also called as Narrow AI, embodies a system designed to carry out one particular job. Weak AI systems include video games such as the chess example from above and personal assistants such as Amazon's Alexa and Apple's Siri. Strong artificial intelligence or Wide AI systems are systems that carry on the tasks considered to be human-like. These tend to be more complex and complicated systems. They are programmed to handle situations in which they may be required to problem solve without having a person intervene. These kinds of systems can be found in applications like self-driving cars or in hospital operating rooms.

2.3 Speech Recognition

Speech recognition, also known as automatic speech recognition, computer speech recognition, or speech-to-text, is a capability which enables a program to process human speech into a written format. While it’s commonly confused with voice recognition, speech recognition focuses on the translation of speech from a verbal format to a text one whereas voice recognition just seeks to identify an individual user’s voice. Many speech recognition applications and devices are available, but the more advanced solutions use AI and machine learning. They integrate grammar, syntax, structure, and composition of audio and voice signals to understand and process human speech. The vagaries of human speech have made development challenging. It’s considered to be one of the most complex areas of computer science involving linguistics, mathematics and statistics. Speech recognizers are made up of a few components, such as the speech input, feature extraction, feature vectors, a decoder, and a word output. The decoder leverages acoustic models, a pronunciation model or pronunciation dictionary, and language models to determine the appropriate output.

Various algorithms and computation techniques are used to recognize speech into text and improve the accuracy of transcription. Below are brief explanations of some of the most commonly used methods:

- **Natural Language Processing**

While NLP isn't necessarily a specific algorithm used in speech recognition, it is the area of artificial intelligence which focuses on the interaction between humans and machines through language through speech and text. Many desktop and mobile devices incorporate speech recognition into their systems to conduct voice search e.g. Siri or provide more accessibility around texting. Natural language processing layers increasingly powerful and complex systems to handle tasks that require higher levels of understanding.

- **Hidden Markov Models**

Hidden Markov Models build on the Markov chain model, which stipulates that the probability of a given state hinges on the current state, not its prior states. While a Markov chain model is useful for observable events, such as text inputs, Hidden Markov Models allow us to incorporate hidden events, such as part-of-speech tags, into a probabilistic model. They are utilized as sequence models within speech recognition, assigning labels to each unit—i.e. words, syllables, sentences, etc.—in the sequence. These labels create a mapping with the provided input, allowing it to determine the most appropriate label sequence.

- **N-grams**

This is the simplest type of language model, which assigns probabilities to sentences or phrases. An N-gram is sequence of N-words. For example, "order the pizza" is a trigram or 3-gram and "please order the pizza" is a 4-gram. Grammar and the probability of certain word sequences are used to improve recognition and accuracy.

- **Neural networks**

Primarily leveraged for deep learning algorithms, neural networks process training data by mimicking the interconnectivity of the human brain through layers of nodes. Each node is made up of inputs, weights, a bias (or threshold) and an output. If that output value exceeds a given threshold, it "fires" or activates the node, passing

data to the next layer in the network. Neural networks learn this mapping function through supervised learning, adjusting based on the loss function through the process of gradient descent. While neural networks tend to be more accurate and can accept more data, this comes at a performance efficiency cost as they tend to be slower to train compared to traditional language models.

- **Speaker Diarization**

Speaker Diarization algorithms identify and segment speech by speaker identity. This helps programs better distinguish individuals in a conversation and is frequently applied at call centers distinguishing customers and sales agents.

2.4 Speech Synthesis

Speech synthesis or text-to-speech abbreviated as TTS, is defined as the artificial production of human voices. The main use is the ability to translate a text into spoken speech automatically. Unlike speech recognition systems that use phonemes in the first place to cut out sentences, TTS will be based on what are known as graphemes: the letters and groups of letters that transcribe a phoneme. This means that the basic resource is not the sound, but the text. This is usually done in two steps. The first will cut the text into sentences and words or graphemes and assign phonetic transcriptions, the pronunciation, to all these groups. Once the different text or phonetic groups have been identified, the second step consists of converting these linguistic representations into sound. In other words, to read these indications to produce a voice that will read the information. In order to choose the right speech synthesis, it is essential to take into account several criteria. These parameters are the following: the language spoken, the type of speaker, the quality of the voice and the supplier. With this information, it is easier to select the right solution that meets user's needs and constraints. Indeed, not all companies offering TTS have equivalent ranges, so it is very important to source these partners well before starting. Next, the language and the type of voice are important criteria for the user experience proposed, there must be consistency between the voice interface and what it should inspire.

Voice assistants like Siri, Google assistant have been directly equipped with voice synthesis in order to be able to respond to the user. This is not insignificant, it is precisely a question of strengthening the relationship between human and machine

through a conversational, reciprocal link. The user talks to the assistant and the latter answers, as in a natural conversation between two or more humans. In fact, like any innovation, the adoption process is generally complex, especially when it brings a break in usage. The best way to gain acceptance for voice assistants was to offer new features that promote their use, but also to improve the user experience as much as possible by humanising the technology. These synthesised voices then made it possible to give an identity to the various assistants, making it possible to differentiate them, but also to consider them as entities in their own right.

2.5 Natural Language Processing

Natural language processing shortened as NLP refers to the branch of computer science and more specifically, the branch of AI concerned with giving computers the ability to understand text and spoken words in much the same way human beings can. NLP combines computational linguistics rule-based modeling of human language with statistical, machine learning, and deep learning models. Together, these technologies enable computers to process human language in the form of text or voice data and to 'understand' its full meaning, complete with the speaker or writer's intent and sentiment. NLP drives computer programs that translate text from one language to another, respond to spoken commands, and summarize large volumes of text rapidly even in real time. Natural language processing includes many different techniques for interpreting human language. Human language is extremely complex and diverse, NLP tasks break down language into shorter, elemental pieces, try to understand relationships between the pieces and explore how the pieces work together to create meaning. Natural language processing layers increasingly powerful and complex systems to handle tasks that require higher levels of understanding.

1. Input and Initial Processing

Taking in speech or text and breaking it up into smaller pieces for processing. For speech, this step is called phonetic analysis, and consists of breaking down the speech into individual sounds, called phonemes. For text input, this can include optical character recognition (OCR) and tokenization. OCR is used to recognize the individual characters in text if it's coming in as an image rather than as words made of characters. Tokenization refers to breaking down a continuous text into individual tokens, often words.

2. **Morphological Analysis**

Breaking down complex words into their components to better understand their meaning. For example, breaking down “incomprehensible” into its component parts.

“in”—not

“comprehens”—to understand or comprehend

“ible”—indicates that this word is an adjective, describing whether something can be comprehended.

3. **Syntactic Analysis**

Trying to understand the structure of sentences by looking at how the words work together. This step is like diagramming a sentence, where you identify the role each word is playing in the sentence.

4. **Semantic Interpretation**

Working out the meaning of a sentence by combining the meaning of individual words with their syntactic roles in the sentence.

5. **Discourse Processing**

Understanding the context around a sentence to fully process what it means.

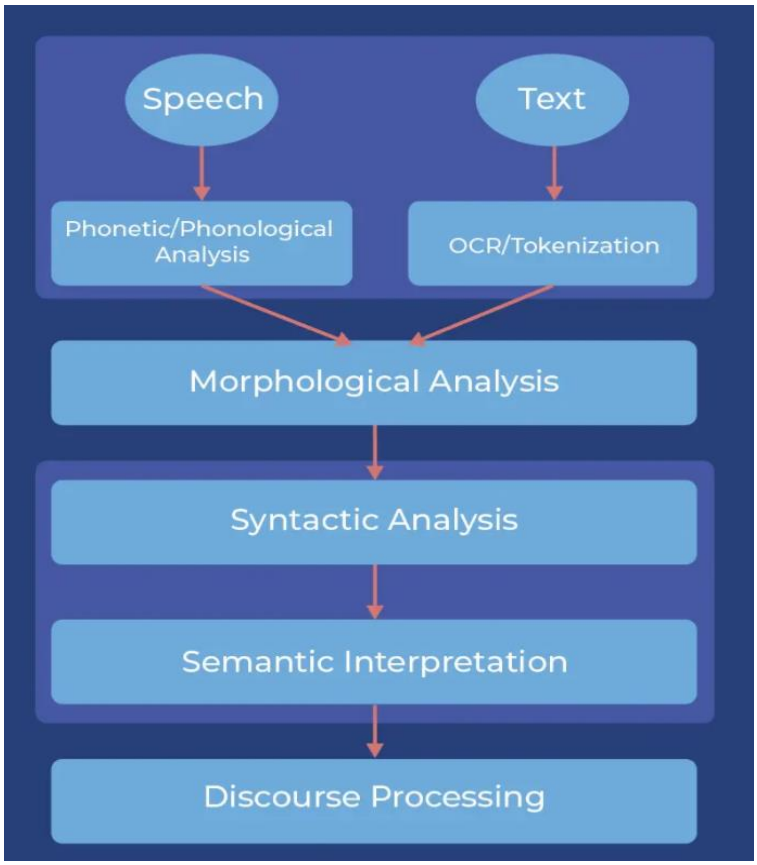


Fig.2.1 Hierarchy of Natural Language Processing

2.4 Aim and Objectives

The aim of proposed system is to design and implement program to build an Intelligent Desktop Voice Assistant.

The objectives of proposed system are:

- To aim the technologies for solving some real-world problems.
- To provide timely and precise results at certain voice commands.
- To give user hands-free access to many functions.
- To save the time by performing mundane tasks more efficiently.

CHAPTER 3
TOOLS AND TECHNOLOGIES

This chapter describes various technologies those are being used in the development of the proposed system. The function and modules are explained along with features and components.

3.1 Python Programming Language

Python is a general purpose, dynamic, high-level, and interpreted programming language. It supports object oriented programming approach as well as functional programming approach. Python's syntax and dynamic typing with its interpreted nature make it an ideal language for scripting and rapid application development. Python supports multiple programming pattern, including object-oriented, imperative, and functional or procedural programming styles. Python is not intended to work in a particular area, such as web programming. That is why it is known as general purpose programming language. They don't need to use data types to declare variable because it is dynamically typed so can be written as a=10 to assign an integer value in an integer variable. Python is interpreted means the Python program is executed one line at a time. The advantage of being interpreted language is that it makes testing and debugging process easy.

3.1.1 Applications of Python

There are so many applications of Python some of them are as follows:

1. **Web Development**

Web framework like Django and Flask are based on Python. They help us to write server side code which helps us to manage database, write backend programming logic, mapping urls etc. Python web frameworks are known for their security, scalability and flexibility.

2. **Machine Learning**

There are many machine learning applications written in Python. Machine learning is a way to write a logic so that a machine can learn and solve a particular problem on its own. For example, voice recognition technology in Apple’s Siri, Google Assistant

etc. uses machine learning algorithm. Face recognition and Recommendation system are another example of machine learning.

3. Game Development

Python comes loaded with many useful extensions that come in handy for the development of interactive games. For instance, libraries like PySoy and PyGame are two Python-based libraries used widely for game development.

4. Scientific and Numeric Applications

Python has become a crucial tool in scientific and numeric computing. In fact, Python provides the skeleton for applications that deal with computation and scientific data processing. Libraries like SciPy, Pandas, Natural Language Toolkit etc. are most useful for scientific and numeric applications.

5. Desktop GUI

Python not only boasts of an English-like syntax, but it also features a modular architecture and the ability to work on multiple operating systems. These aspects, combined with its rich text processing tools, make Python an excellent choice for developing desktop-based GUI applications.

Python offers many GUI toolkits and frameworks that make desktop application development a breeze.

3.1.2 Features of Python

1. Easy to Learn and Use

Python is easy to learn as compared to other programming languages. Its syntax is straightforward and much the same as the English language. There is no use of the semicolon or curly-bracket, the indentation defines the code block.

2. Expressive Language

Python can perform complex tasks using a few lines of code. A simple example is hello world program it take only one line to execute, print(“hello world”).

3. Cross-platform Language

Python can run equally on different platforms such as Windows, Linux, UNIX and Mac etc. Python is a portable language. It enables programmers to develop the software for several competing platforms by writing a program only once.

4. Open Source

Python is freely available for everyone. It is freely available on it official website www.python.org.

5. **Exception Handling**

An exception is an event that can occur during program exception and can disrupt the normal flow of program. Python supports exception handling which means user can write less error prone code and can test various scenarios that can cause an exception later on.

6. **Large Standard Library**

It provides a vast range of libraries for the various fields such as machine learning, web developer, and also for the scripting.

7. **Automatic Memory Management**

Memory allocation can be defined as allocating a block of space in the computer memory to a program. In Python memory allocation and deallocation method is automatic as the Python developers created a garbage collector. Garbage collection is a process in which the interpreter frees up the memory when not in use to make it available for other objects.

3.2 Visual Studio Code IDE

Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including Java, JavaScript, Go, Node.js, Python and C++. It is based on the Electron framework, which is used to develop Node.js Web 1. applications that run on the Blink layout engine. Visual Studio Code employs the same editor component used in Azure DevOps. Instead of a project system, it allows users to open one or more directories, which can then be saved in workspaces for future reuse. This allows it to operate as a language-agnostic code editor for any language. It supports a number of programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via the settings. Many Visual Studio Code features are not exposed through menus or the user interface but can be accessed via the command palette. Visual Studio Code can be extended via extensions, available through a central repository. This includes additions to the editor and language support.

A notable feature is the ability to create extensions that add support for new languages, themes, and debuggers, perform static code analysis, and add code linters using the Language Server Protocol. Visual Studio Code allows users to set the code page in which the active document is saved, the newline character, and the

programming language of the active document. This allows it to be used on any platform, in any locale, and for any given programming language.

Advantages of Visual Studio Code IDE:

- Free, quick and easy to install
- Extensive plugin ecosystem and support
- Extensible in configuration and personalisation
- Intuitive to use
- Personal settings can easily be synced to a personal or group source code management platform

3.3 Python Modules

1. Pyttsx3

pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline and is compatible with both Python 2 and 3. An application invokes the pyttsx3.init() factory function to get a reference to a pyttsx3. Engine instance. it is a very easy to use tool which converts the entered text into speech. The pyttsx3 module supports two voices first is female and the second is male which is provided by "sapi5" for windows. It supports three TTS engines:

- sapi5: SAPI5 on Windows
- nsss: NSSpeechSynthesizer on Mac OS X
- espeak: eSpeak on every other platform

2. Speech_recognition

Speech recognition takes input in form of voice and then convert it to text. speech-recognition is simplest of all the libraries having same purpose. It is capable of recognizing speech from audio files, as well as live from a microphone. Then the speech to text translation is done with the help of Google Speech Recognition. This requires an active internet connection to work. There are several other offline Recognition systems such as PocketSphinx, but have a very rigorous installation process that requires several dependencies. Google Speech Recognition is one of the easiest to use.

3. Datetime

In Python, date and time are not a data type of its own, but a module named datetime can be imported to work with the date as well as time. datetime module comes built into Python. This module supplies classes to work with date and time.

The datetime classes are categorize into 6 main classes:

- **date**: An idealized naive date, assuming the current Gregorian calendar always was, and always will be, in effect. Its attributes are year, month and day.
- **time**: An idealized time, independent of any particular day, assuming that every day has exactly 24*60*60 seconds. Its attributes are hour, minute, second, microsecond.
- **datetime**: Its a combination of date and time along with the attributes year, month, day, hour, minute, second, microsecond, and tzinfo.
- **timedelta**: A duration expressing the difference between two date, time, or datetime instances to microsecond resolution.
- **timezone**: A class that implements the tzinfo abstract base class as a fixed offset.

4. Wikipedia

Python provides the wikipedia module to scrap the data from the Wikipedia pages. This module allows us to get and parse the information from Wikipedia. Wikipedia module consists of various built-in methods which help to get the desired information. These built-in methods include:

- First method is Search title and result. Wikipedia module allows to search a query supplied as an argument using search() method.This method returns list of all articles that contain the search query. The number of search titles can be set to limit by passing the result parameter.
- As the name suggests, suggestion() method returns the suggested Wikipedia title for the query or none if it doesn't found any.
- The summary() method returns the summary of the article or topic. It takes two arguments, title and sentences and returns the summary in the string format.

5. Webbrowser

In Python, webbrowser module provides a high-level interface which allows displaying Web-based documents to users. This module includes functions to open

URLs in interactive browser applications. The module includes a registry of available browsers, in case multiple options are available on the system. The webbrowser module is use to launch a browser in a platform-independent manner. This module opens the requested page using the default browser by simply calling open() function. The documentation says that an existing window will be reused, if possible, but the actual behaviour may depends browser's settings. Different functions can be used to have a bit more control over how the page gets open like opening requested page in new window or new tab.

6. Urllib.request

Urllib module is the URL handling module for Python. It is used to fetch the Uniform Resource Locators. Urllib is a package that collects several modules one of them is urllib.request. It is used for opening and reading purpose of URLs (mostly HTTP). This module helps to define functions and classes for opening purpose.

7. Smtplib

Simple Mail Transfer Protocol (SMTP) is a protocol, which handles sending e-mail and routing e-mail between mail servers. It defines an object known as "SMTP client session object" which is used to send mail by the user. First it initiates Gmail SMTP using smtplib. SMTP(), then identifies the server using ehlo() function, then encrypting the session starttls(), followed by login to mailbox using login() and lastly it sends the message using sendmail().

8. Json

The full-form of JSON is JavaScript Object Notation. It means that a script (executable) file which is made of text in a programming language, is used to store and transfer the data. Python supports JSON through a built-in package called json.we import the json package in Python script. The text in JSON is done through quoted string which contains value in key-value mapping within { }. It is similar to the dictionary in Python and pickle modules and Python natively supports JSON features.

9. PyDictionary

PyDictionary is an open-source dictionary module for Python 2 and Python 3 to get meanings, translations, synonyms and antonyms of words. It uses WordNet for getting meanings, Google for translations, and synonym.com for getting synonyms and antonyms. PyDictionary can be utilised in two ways, either by creating a dictionary instance which can take words as arguments or by creating a dictionary instance with a fixed amount of words.

10. Requests

The requests module in Python allows user to exchange requests on the web. Python requests module has several built-in methods to make Http requests to specified URI using GET, POST, PUT, PATCH or HEAD requests. A Http request is meant to either retrieve data from a specified URL or to push data to a server. HTTP works as a request-response system between a server and a client. Web browser is the client, and the system that hosts the site that user want to access is the server.

- GET method is used to retrieve information from the given server using a given URL. Python's requests module provides in-built method called get() for making a GET request to a specified URL.
- POST request method requests that a web server accepts the data enclosed in the body of the request message, most likely for storing it.
- The PUT method requests that the enclosed entity be stored under the supplied URL. If the URL refers to an already existing resource, it is modified and if the URL does not point to an existing resource, then the server can create the resource with that URL.
- The DELETE method deletes the specified resource.
- The HEAD method asks for a response identical to that of a GET request, but without the response body.
- It is used for modify capabilities. The PATCH request only needs to contain the changes to the resource, not the complete resource.

11. Pyjokes

Python supports creation of random jokes using one of its libraries. Pyjokes is a python library that is used to create one-line jokes for programmers. Informally, it can

also be referred as a fun python library which is pretty simple to use. It returns a random joke from the given category in the given language.

Supported Languages by Pyjokes

- English: ‘en’
- Spanish: ‘es’
- Italian: ‘it’
- German: ‘de’
- Galician: ‘gl’
- Basque: ‘eu’

Categories included in Pyjokes

- For geeky jokes: ‘neutral’ (It is chosen by default)
- For Chris Norris Jokes: ‘chuck’.
- If user want all type of jokes: ‘all’
- There is one more category known as ‘twister’ which only works for the German language(‘de’) and mostly includes tongue twister.

12. OS

The OS module in Python provides functions for interacting with the operating system. OS comes under Python’s standard utility modules. This module provides a portable way of using operating system-dependent functionality. The ‘os’ and ‘os.path’ modules include many functions to interact with the file system. This package abstracts the functionalities of the platform and provides the python functions to navigate, create, delete and modify files and folders. Some functions in the OS module are as follows:

- os.name()

  This function provides the name of the operating system module that it imports.
- os.mkdir()

  The os.mkdir() function is used to create new directory
- os.getcwd()

  It returns the current working directory(CWD) of the file.
- os.chdir()

  The os module provides the chdir() function to change the current working directory.
- os.rmdir()

The rmdir() function removes the specified directory with an absolute or related path.

- os.popen()

  This function opens a file or from the command specified, and it returns a file object which is connected to a pipe
- os.close()

  This function closes the associated file
- os.rename()

  A file or directory can be renamed by using the function os.rename(). A user can rename the file if it has privilege to change the file.
- os.access()

  This function uses real uid/gid to test if the invoking user has access to the path.

13. Pywhatkit

Python offers numerous inbuilt libraries to ease user's work. Among them pywhatkit is a Python library for sending WhatsApp messages at a certain time. kit.sendwhatmsg() is a function that is used to send WhatsApp message. The parameters of this functions are:

- phone_num (required): Phone number of target with country code.
- message (required): Message that user want to sendwhatmsg.
- time_hour (required): Hours at which user want to send message in 24 hour format.
- time_min (required): Minutes at which user want to send message.
- wait_time (optional, val=20): Seconds after which the message will be sent after opening the web.
- print_waitTime (optional, val=True): Will print the remaining time if set to true.

14. Pyautogui

PyAutoGUI is a cross-platform GUI automation Python module. It is used to programmatically control the mouse & keyboard to automate interactions with other applications.

PyAutoGUI has several features:

- Moving the mouse and clicking or typing in the windows of other applications.
- Sending keystrokes to applications (for example, to fill out forms).

- Take screenshots, and given an image (for example, of a button or checkbox), find it on the screen.
- Locate an application's window, and move, resize, maximize, minimize, or close it.
- Display message boxes for user interaction while GUI automation script runs.

15. Wolframalpha

The WolframAlpha Webservice API provides a web-based API allowing the computational and presentation capabilities of WolframAlpha to be integrated into web, mobile, desktop, and enterprise applications. Wolfram Alpha is an API which can compute expert-level answers using Wolfram's algorithms, knowledgebase and AI technology. It is made possible by the Wolfram Language.

16. OpenWeatherMap

OpenWeatherMap provides a range of weather-related products in a variable combination of depth and steps of measurement to millions of clients globally. The product range includes current, historical and forecasted weather data with the granularity as high as 1 minute. The length of the nowcast reaches 2 hours, short-term forecast reaches 16 days and long-term forecast can reach up to 1 year length. Historical weather data goes over 40 years deep. OpenWeather also provides a range of weather maps and weather alert services. OpenWeather provides data for weather risk management on the individual agreement basis to the industries like energy, agriculture, transportation, construction, municipalities, travel, food processors, retail sales and real estate. OpenWeather also operates under the terms of license providing free access to the APIs that include current weather, a minutely forecast for 1 hour, hourly forecast for 48 days, 3-hour forecast for 5 days, daily forecast for 7 days, short-term history, weather maps, alerts, geocoding, air quality weather triggers and weather widgets. The projects with a higher demand of loading, may obtain an extended service on the basis of paid subscription.

CHAPTER 4

PROPOSED APPROACH AND SYSTEM ARCHITECTURE

This chapter represents the architecture of proposed system. It explains the various components and models involve in development of voice assistant in detail. It also describes the approach followed by proposed system.

4.1 Proposed Approach

The proposed system uses the combination of both Task-oriented approach and Knowledge-oriented approach to perform almost every task that user wants to get done.

4.1.1 Task-oriented approach

A task-oriented approach use goals to tasks to achieve what the user needs. This approach often integrates itself with other apps to help complete tasks. For example, if user asks a voice assistant to set an alarm for 3PM, it understand this to be a task request and communicate with the default Clock application to open and set an alarm for 3PM. This approach does not require an extensive online APIs, as it is mainly using the system calls and already existing skills of other installed applications.

4.1.2 Knowledge-oriented approach

A knowledge-oriented approach is the use of analytical data to help users with their tasks. This approach focuses on use of online APIs and already recorded knowledge to help complete tasks. An example of this approach is anytime a user asks for an internet search, it uses the online APIs available to return relevant results and recommend the highest search result. If user searches up a trivia question, this process need knowledge-oriented approach as it is searches for data instead of working with other apps to complete tasks.

4.1.3 Proposed System and Existing System

- One of the main benefit of proposed voice assistant is that the user can personalise the functions of system depending upon the need. User can also control the system's

way of responding back along with the voice that system uses, providing the more human-like conversational environment.

- The proposed system is designed in such a way that user does not have to initiate the process of taking input as voice command, that is, the process of listening, every single time. Once it is initiated the assistant continues to listen the user's command until user asks to stop.
- Accuracy rate in speech recognition and efficiency of proposed system is maximized by using the Google API for its Cloud Service. The Google Cloud provides the access to use vocabulary from a much larger database. It has ability to accept natural speech, different accents and languages.

4.2Architecture of Proposed System

The overall architecture of proposed system consists of following phases:

1. **Speech –to –text**

Taking input from the user in form of voice and converting the speech into text to be processed by the assistant. Voice assistant uses the microphone as source for taking the voice command as the input. This phase then uses speech recognition system for converting the speech to text for further processing.

2. **Command processing and execution**

Next is processing the converted text to get results. The text contains one or two keywords that determine what query is to be executed. If the keyword doesn't match any of the queries in the code, then the assistant asks user to speak again. The proposed system involves the task-oriented as well as knowledge-oriented approach. Thus can easily solve questions using APIs and can perform tasks like opening desktop application using system calls.

3. **Text-to-Speech**

The output which is in form of text is then converted to speech to give final result. This phase uses the speech synthesis system for converting the text to speech. The proposed system not only speaks out result but also displays it on screen.

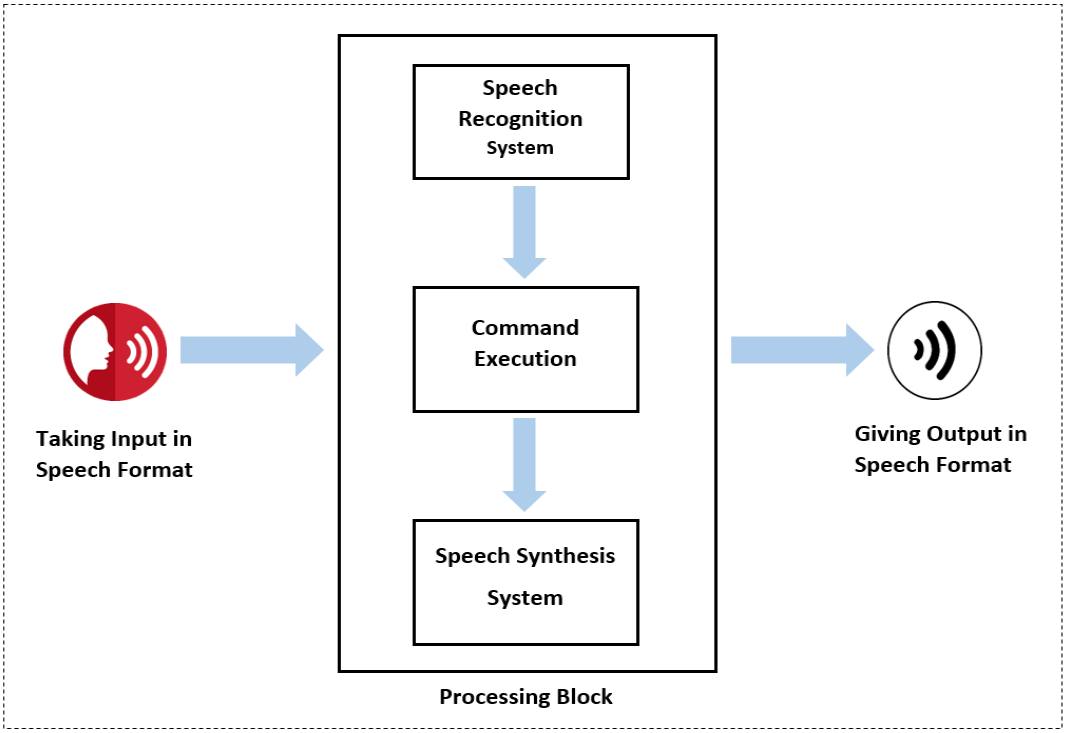


Fig.4.1 Architecture of proposed system

4.3 Architecture of Speech Recognition System

Speech recognition, also known as automatic speech recognition or speech-to-text, is ability of machine which enables a program to process human speech into a written format by identifying the words spoken.

Fig.4.2 Speech-To-Text system

The recognition system has three separate models:

- Acoustic Modeling / Phoneme Detection
- Pronunciation Modeling
- Language Modeling

These three models are trained separately, but are then composed into one gigantic search graph. Essentially, speech recognition is taking an audio waveform, pushing it

through this search graph, and letting it find the path of least resistance—that is, finding the word sequence that has the maximum likelihood. These three models create a huge search graph, through which waveforms can be pushed to create near instantaneous text output.

4.3.1 Acoustic Model

An acoustic model is use to represent the relationship between an audio signal and the phonemes or other linguistic units that make up speech. User speaks to the computer using a microphone. The microphone converts the sound signals into electrical signals. It takes a waveform, chunks it into small time-segments, implements a frequency analysis, and outputs a probability distribution over all the triphone-states for that particular input. The phonemes are the sound units that distinguish one word from the other. The waveform frequency vector, matched with a probability distribution, thus identifies which phonemes are more likely than others to be contained in the audio sample, thereby delivering a sequence of phonemes that exist in that input over time.

Example:

[HOUSE] = hh aw s

4.3.2 Pronunciation Model

A word pronunciation is a possible phoneme-like sequence that can appear in a real utterance and represents a possible acoustic pronunciation of the word. The common dictionary will be created using the words with the accurate phonemes connections called canonical pronunciation. The pronunciation model can be used to map the phoneme-like units to the standard pronunciation that can be found in the common dictionary. These mapping the phoneme units to the correct pronunciation of the word are tougher than the other processes. Basically, it takes the phonemic probability distributions from the acoustic model, checks them against a massive lexicon defining valid sequences of phonemes for the words of a specific language, and restricts the possible phoneme sequences to ones that make sense in that language. That list can’t possibly hold all the valid phonemes of all the words for every language in the world, so the model has a statistical engine which can instantly generate estimated pronunciations based on an orthography for words that it has never seen. It tries to match this with words on the list. These new words typically have a very low

probability score, so they are almost never picked again unless it is the closest match out of all possible words in the lexicon.

Example:

th+is = this

hh+aw+s = house

4.3.3 Language Model

Language modeling is central to many important Natural Language Processing tasks. This is the main step in speech recognition process. Combining the acoustic and pronunciation models, we have audio coming in and words coming out. But that's not quite specific enough to provide reliable Voice Search, because you cannot just string any word together with any other word: there are word combinations that are more reasonable than others. It calculates the frequencies of all word sequences and thereby constrains the possible word sequences that can be formed out of the two aforementioned models to ones that are sensible combinations in language. The final search algorithm will then pick the valid word sequence that has the highest frequency of occurrence in the language. This processes of finding word sequence and understanding the meaning of predicted words and then choosing the most appropriate meaning is nothing but the Natural Language Processing. The model learns about the probability distributions of a sentence over the sequences of words, that is, the likelihood of different phrases in a sentence.

P(w) = P (w1,w2,w3,…,wn)

Example:

P(“this is a house”)= P(this) * P(is|this) * P(a|this is)* P(house|this is a)

It provides the context to distinguish between words and phrases that sound similar.

“recognize speech”, “wreck a nice beach”

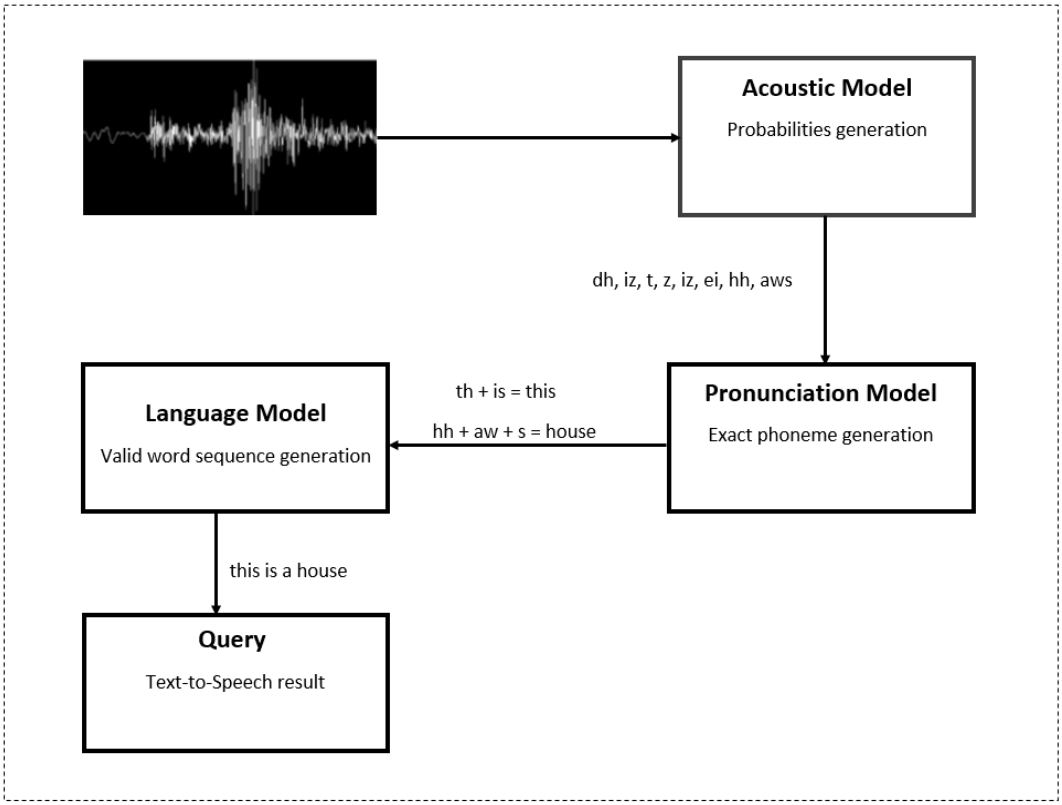


Fig.4.3 Architecture of speech recognition system

4.4 Architecture of Speech Synthesis System

Speech synthesis is the artificial production of human speech or text-to-speech system converts normal language text into speech. Here, the input is text and output is speech.

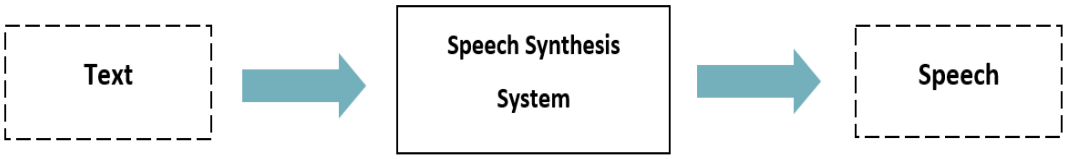


Fig.4.4 Text-to-Speech system

A text-to-speech system is composed of two parts:

- Natural Language Processing Module
- Digital Signal Processing Module

NLP module has two major tasks, firstly, it converts raw text containing symbols, such as numbers and abbreviations, into the equivalent of written words. This process is often called text normalization or text analysis. This process then assigns phonetic

transcriptions to each word, and divides and marks the text into prosodic units, like phrases, clauses, and sentences. The process of assigning phonetic transcriptions to words is called text-to-phoneme or grapheme-to-phoneme conversion. Phonetic transcriptions and prosodic information together make up the symbolic linguistic representation that gives output of NLP module. Digital Signal Processing module often referred to as the synthesizer, then converts the symbolic linguistic representation into sound. Text-to-speech synthesis takes place in several steps. The TTS systems gets a text as input, which it first must analyse and then transforms into a phonetic description. Then in a further step it generates the prosody. From the information now available, it can produce a speech signal.

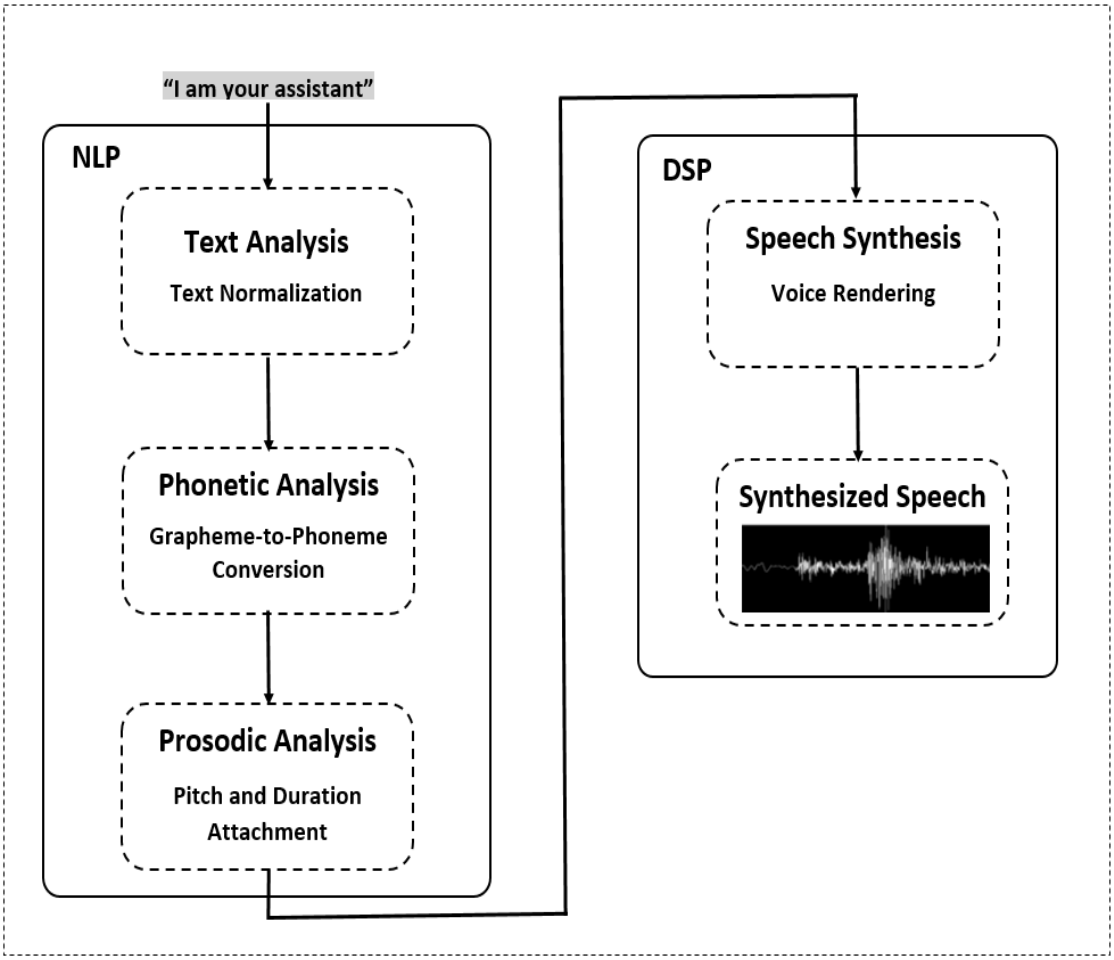


Fig.4.5 Architecture of speech synthesis system

4.4.1 Natural Language Processing Module

It produces a phonetic transcription of the text read, together with prosody. The major operations of the NLP module are as follows:

- **Text Analysis or Text Normalisation**

All written languages have special constructs that require a conversion of the written form or orthographic form into the spoken form. First the text gets segmented into tokens. The token-to-word conversion creates the orthographic form of the token. For the token “Mr” gives the orthographic form “Mister” by expansion, the token “12” gives the orthographic form “twelve”, for “$200” it says “200 hundred dollars” and “1997” transforms to “nineteen ninety-seven”.

- **Phonetic Analysis**

Phonetic analysis or grapheme-to-phoneme conversion or text-to-phoneme conversion is the process of generating pronunciation for words based on their written form. The spelling of a word is called a grapheme and the phonetic form is called phonemes. Speech synthesis systems uses two basic approaches to determine the pronunciation of a word based on its spelling. The simplest approach to text-to-phoneme conversion is the dictionary-based approach, where a large dictionary containing all the words of a language and their correct pronunciations is stored in program. The other approach is rule-based, in which pronunciation rules applied to words determines their pronunciations based on their spellings. This is similar to the synthetic phonics, approach to learning reading. Proposed system uses combination of these approaches for getting more accurate result and handling exception.

- **Prosodic Analysis**

Prosody is the set of features of speech output that includes the pitch also called intonation, the timing, the pausing, the speaking rate, the emphasis on words and many other features. Producing human-like prosody is important for making speech sound natural and for correctly conveying the meaning of spoken language. This can be achieved through analysis of the document structure, sentence syntax, and other information that can be inferred from the text input.

4.4.2 Digital Signal Processing Module

It transforms the symbolic information it receives from NLP into audible and intelligible speech.

- **Waveform Generation**

Its task is to select the appropriate sequence of units from the inventory; modify the pitch, duration of each unit; and concatenate these modified units to produce the desired speech waveform. Thus, the final output of this step is speech.

CHAPTER 5
IMPLEMENTATION

It represents the development stages of proposed system. This chapter also describes the functionalities and testing methods for proposed voice assistant. The implementation of system takes place in three phases as follows:

5.1Developing the conversational environment

For the development of conversational environment, the first and foremost thing is to make Voice assistant talk, the engine is set to Pyttsx3 which is Python text to speech module and Microsoft Speech API sapi5 uses this for speech synthesis. For the proposed voice assistant the female voice is used by setting the voice.id[1].

```
engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
engine.setProperty('voice', voices[0].id)
```

Fig.5.1 Setting voice for assistant

```
def speak(audio):
    engine.say(audio)
    engine.runAndWait()
    print(audio)
```

Fig.5.2 Function to make assistant speak

The next functionality that proposed voice assistants offers, is taking voice command as input with the help of microphone of the system. The function take_command() uses speechRecognition module which supports Google Cloud Speech API for converting speech into text format. A try and except block is also added to the program to handle the errors effectively.

```
def take_command():
    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("Listening...")
        r.pause_threshold = 1
        audio = r.listen(source)

    try:
        print("Recognizing...")
        query = r.recognize_google(audio, language='en-IN')
        query = query.lower()
        print(f"User said: {query}\n")

    except Exception as e:
        print(e)
        speak("say that again please...")
        return "None"
    return query
```

Fig. 5.3 Function to capture voice command

5.2Developing code for handling commands

The Python script for handling various user commands is developed. Decision making is required for executing code only if a certain condition is satisfied. Here, in the implemented code condition satisfaction is based on checking if the keyword of the query is present in the code. Execution occurs only when the query is found to be present in the code. Depending upon the functionalities that user want in voice assistant the python script is developed for the same. The functionalities developed for proposed system are:

Functionality	Description
Greeting user	Greets user depending upon the time .i.e. if time is 8 am, system greets user “Good morning”
Getting date and time	Tells user the current time and today’s date depending on request.
Getting weather report	Gives weather report with temperature and weather description of the city which user asked for.
Getting news report	Opens the preferred news website as per user’s choice and shows the headline for that day.
Getting current location	Accesses the current location of user with the city and country name by using IP address .
Browsing Google	Searches any query that user want access just on one voice command on Google web browser.

Accessing YouTube	Opens YouTube and also plays the song depending upon user's choice.
Accessing Wikipedia	Accesses the information about the topic that user request for and then reads it for user.
Sending E-mail	Generates the complete email just by taking the content from voice command and sends it.
Sending WhatsApp text	Takes the message from user as voice command and then sends message as per preferred time set by user.
Launching and closing applications	Launches any application that user want to use and can also close it, if requested.
Using dictionary	Finds meaning of any word in its noun form and also in adjective form.
Solving calculations	Solves the mathematical calculation just by question from the user as voice input.
Switching tabs	Switches between the open tabs without stopping the ongoing processing.
Shutting down the system	Shutdowns the system after certain time interval set by user.

5.3Testing and Debugging

- **Functional Testing**

Each functionality of the voice assistant is tested with the set of questions to verify that the desired output for each question is obtained.

- **Stress Testing**

In this testing method the same functionality is tested against all the possible command with minimal keywords that user can pass for getting certain result.

- **Validation Testing**

The system's ability of choosing most valid or appropriate word is tested against the homophones, the words with same pronunciation but different spelling. This is done to see if system can differentiate between homophones by extracting the context from user's command.

In debugging stage, the error handling procedures are introduced to prevent the system failing while executing commands.

CHAPTER 6
RESULT AND DISCUSSION

In this chapter, the various results of execution are shown. When the system runs the program, the following results are displayed. All the preparation had been done and the testing was performed according to the various types of input provided and got outputs. Various inputs lead to the unique outputs.

6.1 Result for Functional Commands

- **Getting Date and Time**

```
Good Morning!
I am your assistant, Samantha
Please tell me how can I help you?
Listening...
Recognizing...
User said: what is time

The current time is 01 hour:54 minute:41 second
Listening...
Recognizing...
User said: what is date

Today's date  is 2021-06-05
```

Fig.6.1 Time and Date functionality

In the time functionality current time is displayed and spoken by system in format(HH/MM/SS). Similarly, when date is asked it returned today's date in format (YYYY/MM/DD).

- **Getting Weather Report**

```
Listening...
Recognizing...
User said: show me weather report

getting the API key
For which city you want me too find the weather?
Listening...
Recognizing...
User said: mumbai

Showing you the weather report
 The temperature in Kelvin is 301.14 The humidity is 78 and The weather description is haze
```

Fig.6.2 Weather functionality

In this functionality, to get the real-time weather update, weather API is used. The API key is used for accessing the weather API account. It again asked user the name of city for which user want to find the weather, after taking it as input the result is both displayed and spoken by voice assistant. The result included temperature, humidity and weather description.

- **Getting News Report**

```
Listening...
Recognizing...
User said: show me the news report

taking you to today's news headlines
```

Fig.6.3 News functionality

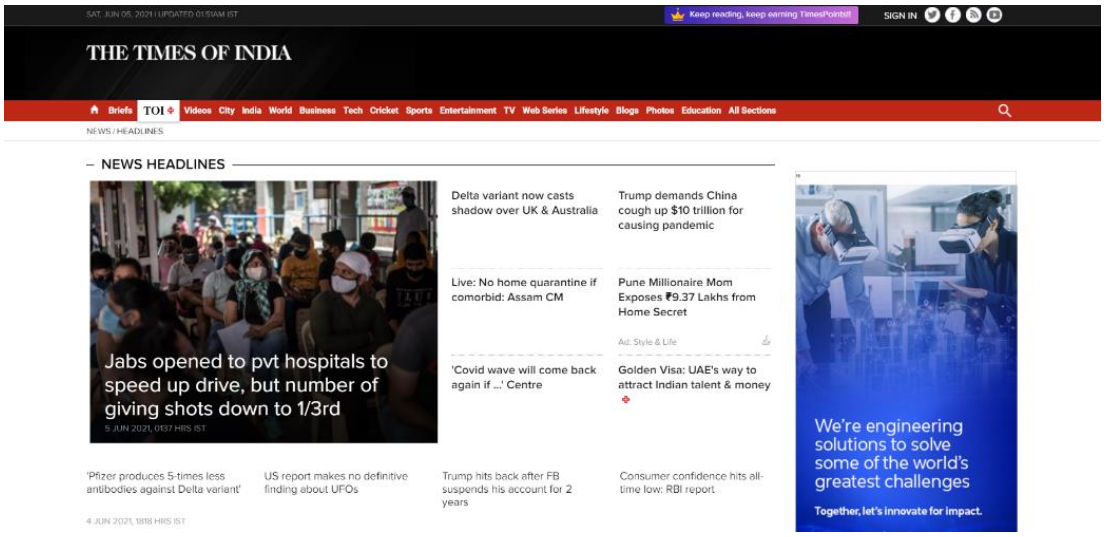


Fig. 6.4 Result of news functionality

On the user request voice assistant opened the news website showing all headlines for current day.

- **Getting Current Location**

```
Listening...
Recognizing...
User said: what is my current location
Getting IP address for finding the location
you are currently in Pune city of country India
```

Fig.6.5 Location functionality

The current location of user is accessed by using the IP address. The output returned both city name as well as country name in which user is currently located.

- **Browsing Google**

```
Listening...
Recognizing...
User said: open google
What should I search on google

Listening...
Recognizing...
User said: what is quantum computing
```

Fig.6.6 Browsing functionality

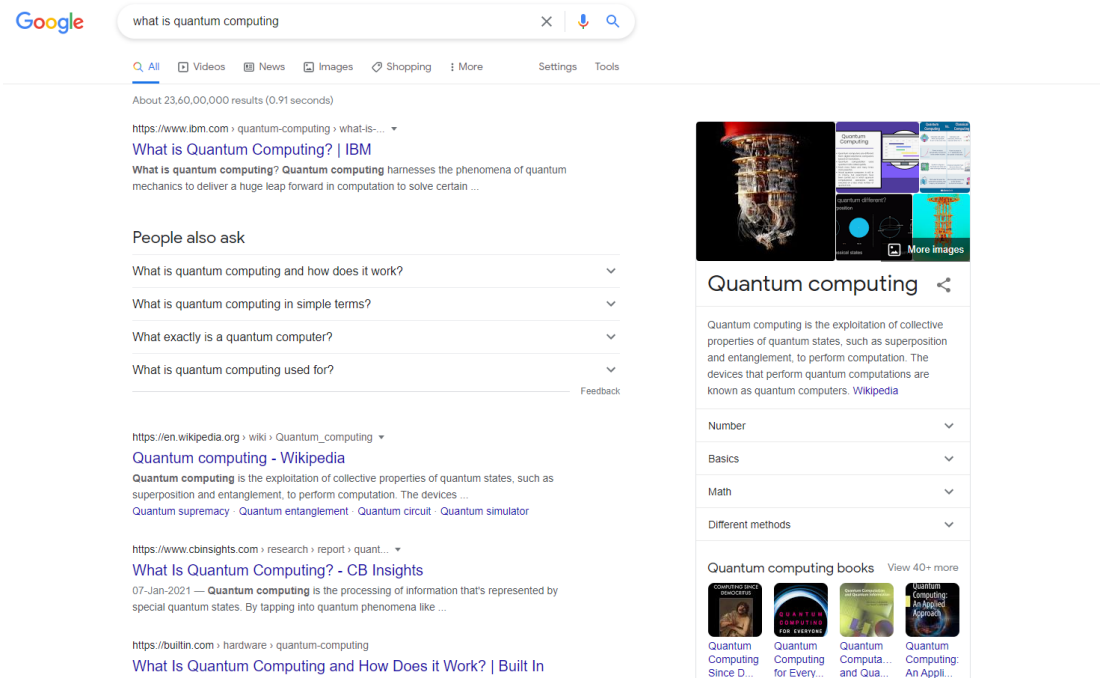


Fig. 6.7 Result of browsing functionality

This functionality not only opened the Google chrome but also searched for query that user asked. This process provided the user hands-free access right from start to getting search results.

- **Accessing YouTube**

```
Listening...
Recognizing...
User said: play happier song on youtube
playing  happier song on youtube
```

Fig.6.8 YouTube functionality

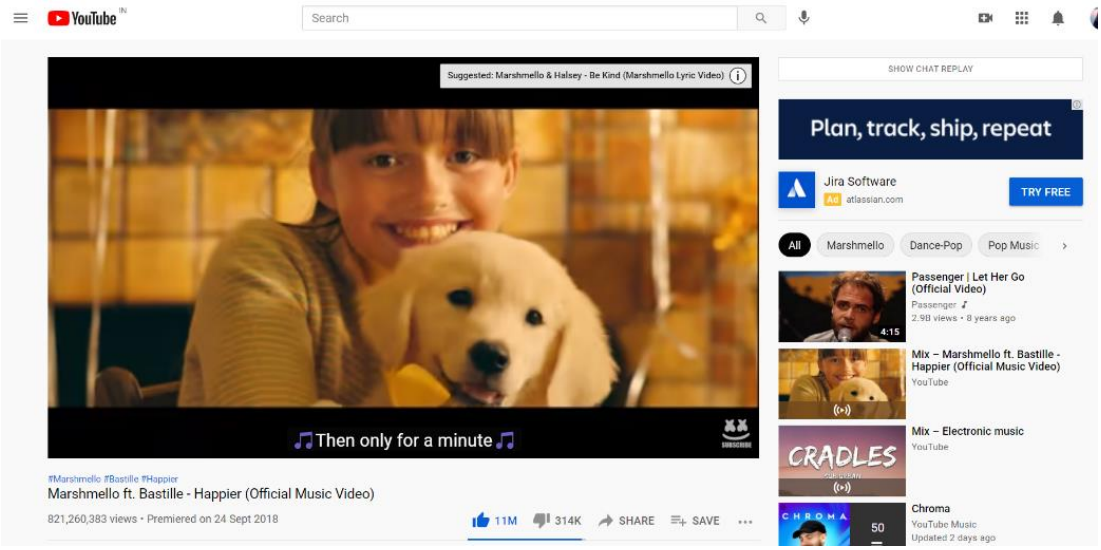


Fig 6.9. Result of YouTube functionality

The voice assistant opened the YouTube and started playing the song that user wanted to play.

- **Accessing Wikipedia**

```
Listening...
Recognizing...
User said: search for elon musk on wikipedia

Searching Wikipedia...
According to Wikipedia
Elon Reeve Musk  ( EE-lon; born June 28, 1971) is an entrepreneur and business magnate. He is the founder, CEO, and chief engineer at SpaceX;
```

Fig. 6.10 Wikipedia functionality

The system used Wikipedia for searching and getting the result of user's query. The result is summarized to first 3 sentences of every topic.

- **Sending E-mail**

```
Listening...
Recognizing...
User said: send email to alexa

What should I say?
Listening...
Recognizing...
User said: this mail is sent by voice assistant

Email has been successfully sent!
```

Fig.6.11 E-mail functionality

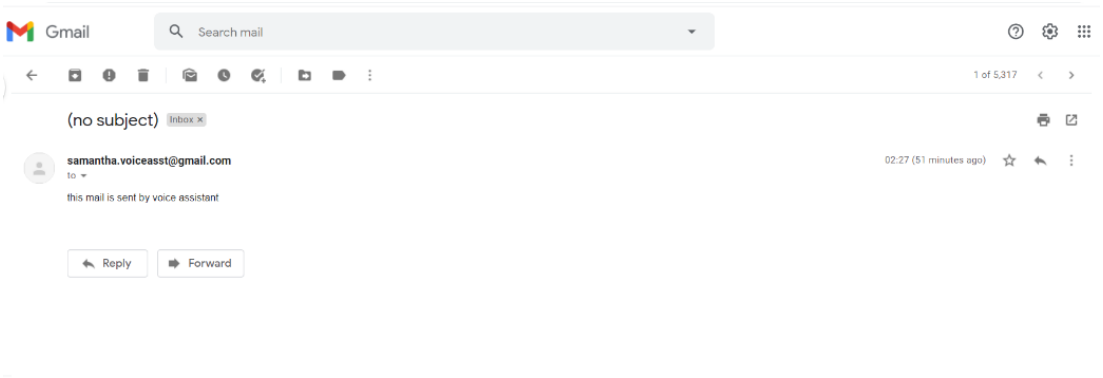


Fig. 6.12 Result E-mail functionality

For this functionality, voice assistant used the parameters like sender's mail id, receiver's mail id and password. The user is then asked for content of the mail. The content is then included in the mail and sent to receiver.

- **Sending WhatsApp Message**

Fig. 6.13 Message functionality

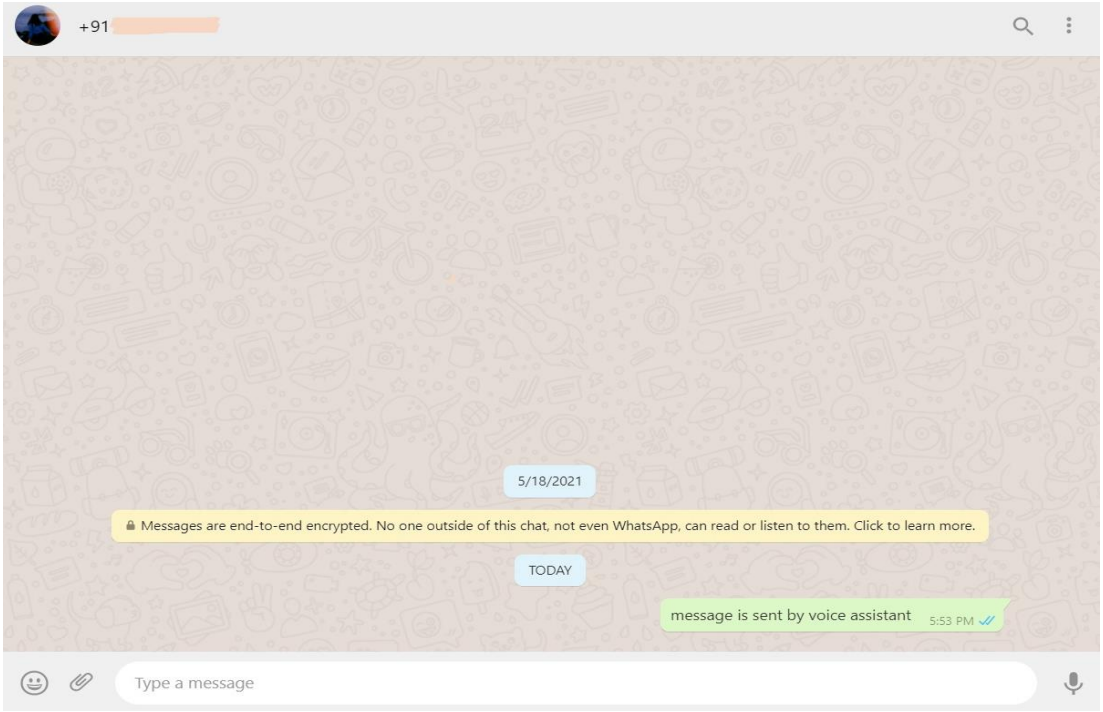


Fig. 6.14 Result of message functionality

This functionality used two parameters that are mobile number and time after which the user want to send this message. Voice assistant asked user to give the message content. The web browser opened WhatsApp and created the message and then sent it to user after certain time.

- **Launching and Closing Applications**

```
Listening...
Recognizing...
User said: open notepad

opening notepad
```

Fig.6.15 Notepad (opening) functionality

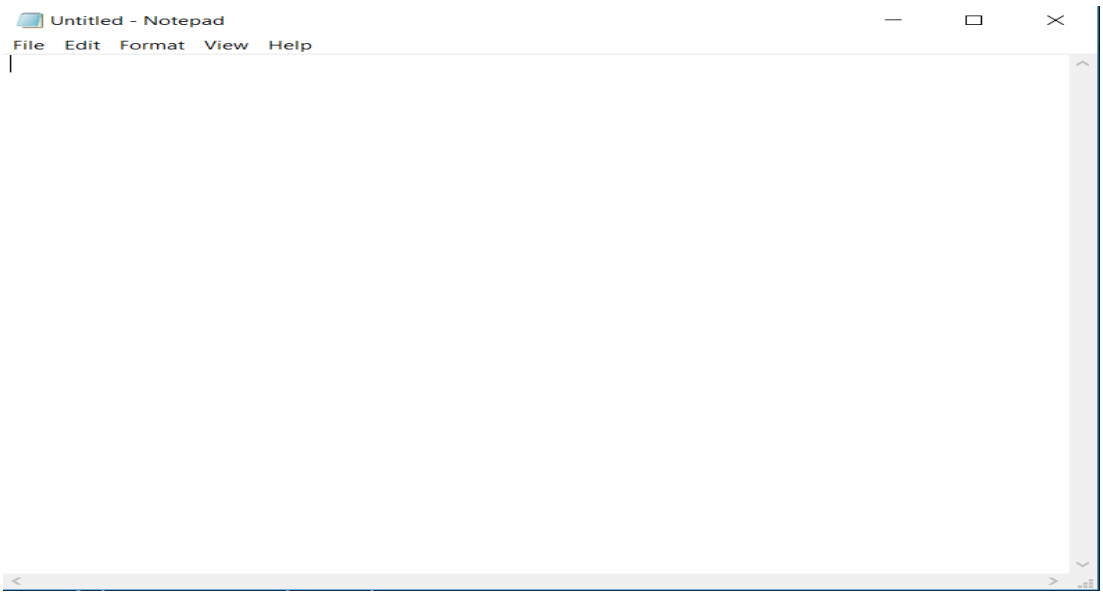


Fig. 6.16 Result of Notepad (opening) functionality

```
Listening...
Recognizing...
User said: close notepad

Closing the notepad
SUCCESS: The process "notepad.exe" with PID 8036 has been terminated.
```

Fig. 6.17 Notepad (closing) functionality

The desktop application Notepad is launched when user commanded. On receiving the user's command to close the launched application, voice assistant closed that.

- **Using Dictionary**

```
Listening...
Recognizing...
User said: open dictionary
what word should I look for?

Listening...
Recognizing...
User said: kind
Alright here is the information you asked for
when kind is used as a Noun then the meanings are
a category of things distinguished by some common characteristic or quality

when kind is used as a Adjective then the meanings are
having or showing a tender and considerate and helpful nature; used especially of persons and their behavior
agreeable, conducive to comfort
tolerant and forgiving under provocation
```

Fig.6.18 Dictionary functionality

In the dictionary functionality, the voice assistant responded with the meaning of the required word in its noun form and also in adjective form.

- **Solving Calculations**

```
Listening...
Recognizing...
User said: calculate
what should I calculate

Listening...
Recognizing...
User said: sin x dx
the answer is
integral sin(x) dx = -cos(x) + constant

Listening...
Recognizing...
User said: do calculation
what should I calculate

Listening...
Recognizing...
User said: 897 x 534
the answer is
478998
```

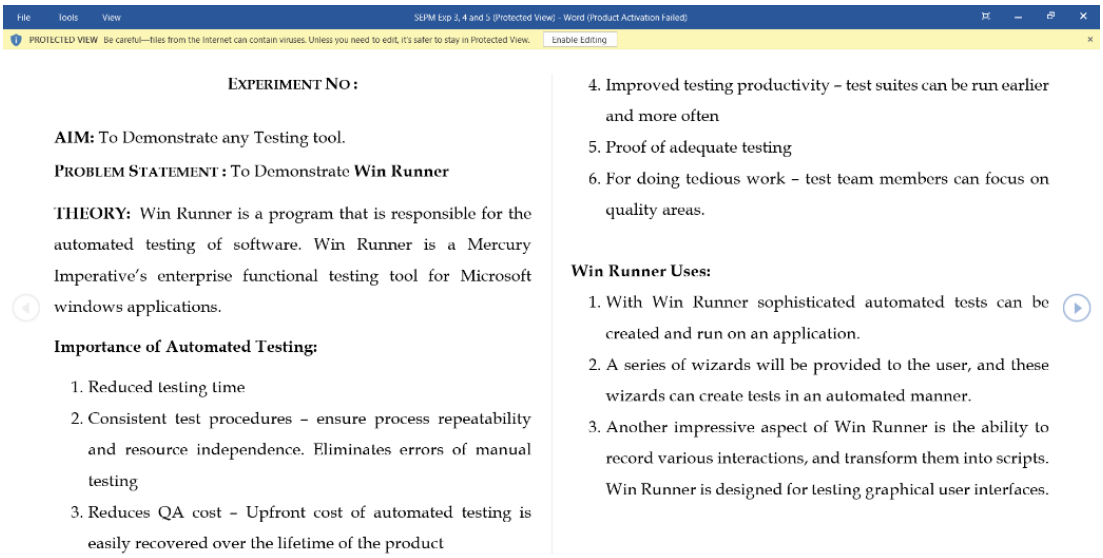
Fig. 6.19 Calculation functionality

This functionality used the Wolframalpha API for solving the mathematical questions. The API used the key for accessing the account and then returned the result accurately. It solved not only multiplication but also the integral of trigonometric function.

- **Switching Tabs**

```
Listening...
Recognizing...
User said: switch tab
Switching the tab
```

Fig.6.20 Switch tab functionality



EXPERIMENT NO :

AIM: To Demonstrate any Testing tool.

PROBLEM STATEMENT : To Demonstrate **Win Runner**

THEORY: Win Runner is a program that is responsible for the automated testing of software. Win Runner is a Mercury Imperative's enterprise functional testing tool for Microsoft windows applications.

Importance of Automated Testing:

1. Reduced testing time
2. Consistent test procedures – ensure process repeatability and resource independence. Eliminates errors of manual testing
3. Reduces QA cost – Upfront cost of automated testing is easily recovered over the lifetime of the product
4. Improved testing productivity – test suites can be run earlier and more often
5. Proof of adequate testing
6. For doing tedious work – test team members can focus on quality areas.

Win Runner Uses:

1. With Win Runner sophisticated automated tests can be created and run on an application.
2. A series of wizards will be provided to the user, and these wizards can create tests in an automated manner.
3. Another impressive aspect of Win Runner is the ability to record various interactions, and transform them into scripts. Win Runner is designed for testing graphical user interfaces.

Fig. 6.21 Result of switch tab functionality

The switch functionality switches between the open tabs. On switch command of user, the voice assistant switched from active tab to the next open tab. The user can go back to the previously active tab by switch command again.

- **Shutting Down the System**

```
Listening...
Recognizing...
User said: shutdown the system
```

Fig. 6.22 Shutdown functionality

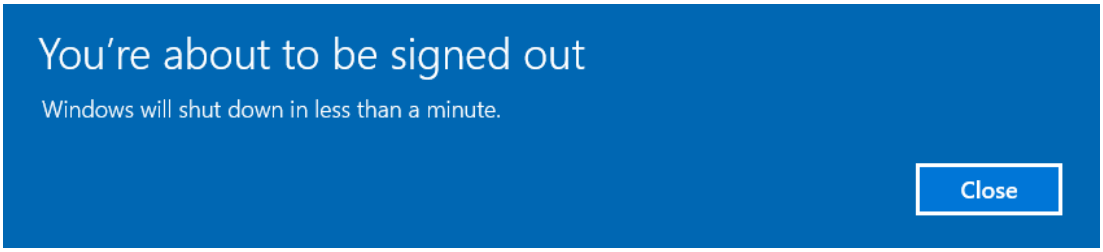


Fig.6.23 Shutdown notification

The assistant first generated a notification before shutting down system. Then within the one-minute system is shut downed.

6.2 Result for Multiple Command Handling

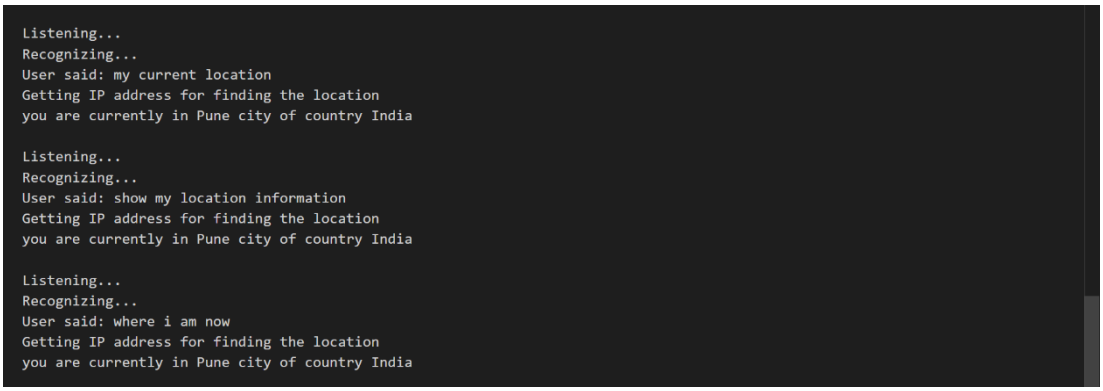


Fig. 6.24 Multiple command handling

Multiple commands are passed for getting the same accurate result. This is done to ensure that if the system can easily understand what user want to ask even when the user's commands vary for the same function. Here for "my current location", "show my location information" and "where am I now" commands the system accessed the location of user accurately.

6.3 Result for Validation

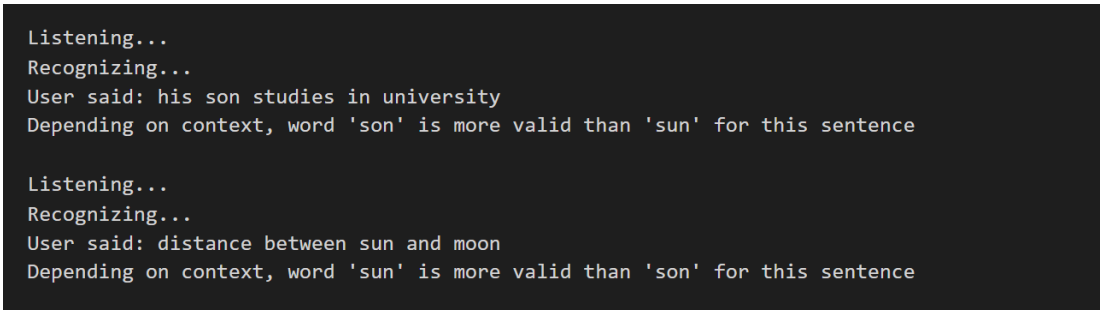


Fig. 6.25 Validation using NLP

Accuracy of speech recognition is main factor in determining the efficiency of voice assistant. It is important to check if system can differentiate between the same sounding words but having different meanings. So, for ensuring the accurate working it is important to know that if system is really using the NLP to understand the context. The word “son” and “sun” are homophones, words with same pronunciation and different spelling and meaning. Now when user said “his son studies in university”, the system determined the probability of words “son” and “sun”, then the word having maximum probability with the given context is selected. The ability to differentiate between the homophones and using most appropriate one based on context, maximizes the accuracy rate of overall system.

CHAPTER 7
CONCLUSION

This system is designed in such a method wherein the user accommodates to it effortlessly. The proposed system is implemented using speech recognition and speech synthesis. The proposed intelligent desktop voice assistant throws light on word recognition problem when homophones are used. It uses Natural Language Processing for recognizing the most appropriate word which user said by understanding the context. This makes system more reliable and easily accessible to user. In beginning, the current voice assistants and main issues behind it are introduced. After that the related work in field of voice assistant is covered. In the most crucial part, comprehensive amount of study is done about the overall system architecture along with task-oriented and knowledge-oriented approach.

The system also operates in background thus, establishing hands-free interface between user and desktop. Moreover, the system carries out variety of tasks with ease such as telling date and time, playing videos, telling some jokes, getting weather and news report, getting current location, solving calculations, sending e-mails, sending WhatsApp message, finding meaning of any word, answering any user query, switching between the tabs, opening/closing desktop application and even shutting down a user's desktop, just on simple voice command.

Thus, it can conclude that the proposed intelligent desktop voice assistant can effectively perform various tasks on voice commands in less time.

7.1 Limitations of the Study

There are three prerequisite that system should meet for smooth work experience.

1. The working microphone is necessary for capturing user's voice command.
2. The stable internet connection is must.
3. The user must have unique API keys for accessing various APIs.

7.2 Future Scope of Work

The voice assistant system in near future can be improved in few ways to make it more usable and accessible.

1. **Ensuring Offline Working**

At remote places where a reliable internet connection may be not be available all the time, the developed system becomes difficult to be used. To overcome this, the offline speech recognition system can be implemented. This will make system more reliable.

2. **Expanding the Scope**

The system is currently developed to perform the various tasks depending on user requirement. It can potentially be developed for any sector or business based on their requirements for performing specific tasks more efficiently.

3. **Introducing Multiple Language Support**

The system is currently usable for the English language. However, this language support can be improved by including the different languages. This will make it possible for user to access the voice assistant in native language giving user more friendly environment.

References

1. Prerna Wadikar, Nidhi Sargar, Rahool Rangnekar, Prof.Pankaj Kunekar (2020) "Home Automation using Voice Commands in the Hindi Language", *International Research Journal of Engineering and Technology (IRJET)*
2. Chen-Yen Peng and Rung-Chin Chen (2018) "Voice Recognition by Google Home and Raspberry Pi for Smart Socket Control", *Tenth International Conference on Advanced Computational Intelligence (ICACI)*
3. Steve Joseph, Chetan Jha, Dipesh Jain, Saurabh Gavali, Manish Salvi (2020) "Voice based E-Mail for the Blind", *International Research Journal of Engineering and Technology (IRJET)*
4. Nishank Tembhurne, Sumedh Vaidya, Afrin Shiekh, Prof. Swapnil Dravyakar (2019) "Voice Assistant for Visually Impaired People", *International Research Journal of Engineering and Technology (IRJET)*
5. Shen Hui, Song Qunying and Andreas Nilsson (2012**)** "Intelligent Voice Assistant", *DiVA Portal*
6. Harkishen Singh, Muskan Khedia, Jayashree Panda, Subham Mishra, Ankit Singh (2020) "Jarvis: The Personal Linux Assistant"
7. Rishabh Shah, Siddhant Lahoti, Prof. Lavanya. K (2017) "An Intelligent Chatbot using Natural Language Processing", *International Journal of Engineering Research*, Vol. 6, pp. 281-286
8. Win Shih and Erin Rivero (2020) "Virtual Voice Assistants", *Library Technology Reports* vol. 56, no. 4
9. Khushitha Anand, Prince, Jithin P Sajeevan (2020) "Voice Assistant for Windows", *International Journal for Scientific Research & Development (IJSRD)* Vol 8
10. Nikhil Patel, Trupti Landge, Radhika Tiwari, Arjun Verma, Prof. Shabana Pathan (2020) "Desktop Voice Assistant", *International Research Journal of Engineering and Technology (IRJET)*
11. George Terzopoulos and Maya Satratzemi (2019) "Voice Assistants and Artificial Intelligence in Education", *The 9th Balkan Conference*
12. https://ieeexplore.ieee.org/document/9051160 Accessed on 1st April 2021 at 2:21 p.m.
13. https://towardsdatascience.com/understanding-nlp-how-ai-understands-our-languages-77601002cffc Accessed on 4th April 2021 at 3:42 p.m.